Network Caching Technologies

Although the volume of Web traffic on the Internet is staggering, a large percentage of that traffic is redundant—multiple users at any given site request much of the same content. This means that a significant percentage of the WAN infrastructure carries the identical content (and identical requests for it) day after day. Eliminating a significant amount of recurring telecommunications charges offers an enormous savings opportunity for enterprise and service provider customers.

Web caching performs the local storage of Web content to serve these redundant user requests more quickly, without sending the requests and the resulting content over the WAN.

Growth of Web Content

Data networking is growing at a dizzying rate. More than 80% of Fortune 500 companies have Web sites. More than half of these companies have implemented intranets and are putting graphically rich data onto the corporate WANs. The number of Web users is expected to increase by a factor of five in the next three years. The resulting uncontrolled growth of Web access requirements is straining all attempts to meet the bandwidth demand.

Caching

Caching is the technique of keeping frequently accessed information in a location close to the requester. A Web cache stores Web pages and content on a storage device that is physically or logically closer to the user—this is closer and faster than a Web lookup. By reducing the amount of traffic on WAN links and on overburdened Web servers, caching provides significant benefits to ISPs, enterprise networks, and end users. There are two key benefits:

- *Cost savings due to WAN bandwidth reduction*—ISPs can place cache engines at strategic points on their networks to improve response times and lower the bandwidth demand on their backbones. ISPs can station cache engines at strategic WAN access points to serve Web requests from a local disk rather than from distant or overrun Web servers.

  In enterprise networks, the dramatic reduction in bandwidth usage due to Web caching allows a lower-bandwidth (lower-cost) WAN link to serve the same user base. Alternatively, the organization can add users or add more services that use the freed bandwidth on the existing WAN link.

- *Improved productivity for end users*—The response of a local Web cache is often three times faster than the download time for the same content over the WAN. End users see dramatic improvements in response times, and the implementation is completely transparent to them.

Other benefits include the following:

- *Secure access control and monitoring*—The cache engine provides network administrators with a simple, secure method to enforce a sitewide access policy through URL filtering.
- *Operational logging*—Network administrators can learn which URLs receive hits, how many requests per second the cache is serving, what percentage of URLs are served from the cache, and other related operational statistics.

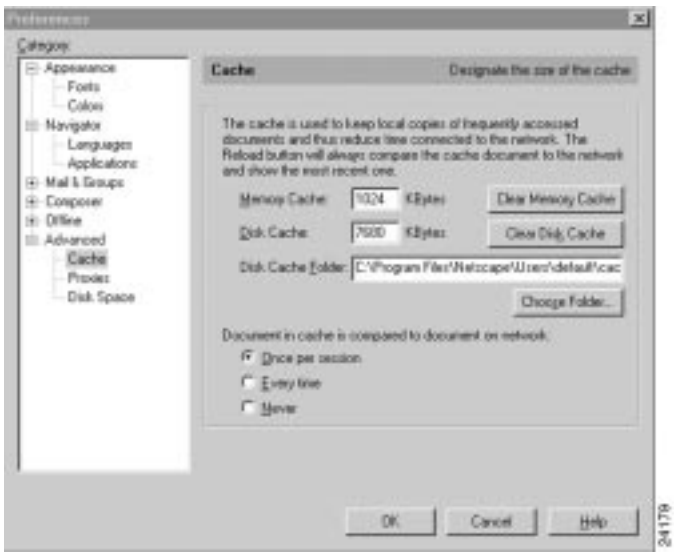
Web caching works as follows:

1. A user accesses a Web page.
2. While the page is being transmitted to the user, the caching system saves the page and all its associated graphics on a local storage device. That content is now cached.
3. Another user (or the original user) accesses that Web page later in the day.
4. Instead of sending the request over the Internet, the Web cache system delivers the Web page from local storage. This process speeds download time for the user, and reduces bandwidth demand on the WAN link.
5. The important task of ensuring that data is up-to-date is addressed in a variety of ways, depending on the design of the system.

Browser-Based Client Caching

Internet browser applications allow an individual user to cache Web pages (that is, images and HTML text) on his or her local hard disk. A user can configure the amount of disk space devoted to caching. Figure 49-1 shows the cache configuration window for Netscape Navigator.

Figure 49-1 You use the cache configuration window to configure the amount of disk space devoted to caching in Netscape Navigator.

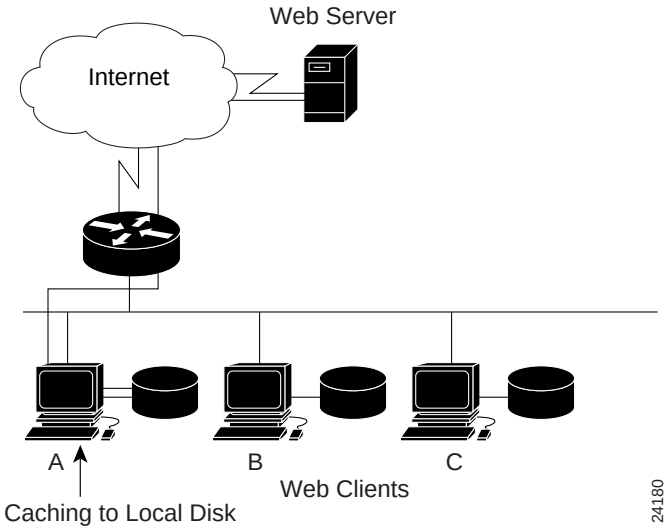


This setup is useful in cases where a user accesses a site more than once. The first time the user views a Web site, that content is saved as files in a subdirectory on that computer's hard disk. The next time the user points to this Web site, the browser gets the content from the cache without accessing the

network. The user notices that the elements of the page—especially larger Web graphics such as buttons, icons, and images—appear much more quickly than they did the first time the page was opened.

This method serves this user well, but does not benefit other users on the same network who might access the same Web sites. In Figure 49-2, the fact that User A has cached a popular page has no effect on the download time of this page for Users B and C.

Figure 49-2 This figure demonstrates the benefits gained by a single node using browser caching.



Attempts at a Caching Solution on the Network Level

To limit bandwidth demand caused by the uncontrolled growth of Internet use, vendors have developed applications that extend local caching to the network level. The two current types of network-level caching products are proxy servers and network caches:

- *Proxy servers* are software applications that run on general-purpose hardware and operating systems. A proxy server is placed on hardware that is physically between a client application, such as a Web browser, and a Web server. The proxy acts as a gatekeeper that receives all packets destined for the Web server and examines each packet to determine whether it can fulfill the requests itself; if it can't, it forwards the request to the Web server. Proxy servers can also be used to filter requests, for example, to prevent employees from accessing a specific set of Web sites.

  Unfortunately, proxy servers are not optimized for caching, and they fail under heavy network loads. In addition, because the proxy is in the path of all user traffic (it's a "bump in the cable"), two problems arise: All traffic is slowed to allow the proxy to examine each packet, and failure of the proxy software or hardware causes all users to lose network access. Further, proxies require configuration of each user's browser—an unacceptable option for service providers and large enterprises. Expensive hardware is required to compensate for low software performance and the lack of scalability of proxy servers.

- In response to these shortcomings, some vendors have created *network caches*. These caching-focused software applications are designed to improve performance by enhancing the caching software and eliminating the other slow aspects of proxy server implementations. However, because network caches run under general-purpose operating systems (such as UNIX

or Windows NT) that involve very high per-process context overhead, they cannot scale to large numbers of simultaneous processes in a graceful fashion. This is especially true for networking caching systems that can have many thousands of simultaneous and short-lived transactions.

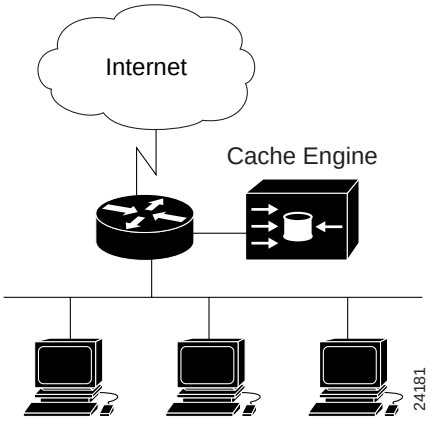
Cisco's Network-Based Shared Caching

The cache engine was designed from the ground up as a loosely coupled, multinode network system optimized to provide robust shared network caching. The cache engine solution comprises the Web Cache Control Protocol (a standard feature of Cisco IOS software) and one or more Cisco cache engines that store the data in the local network.

The Web Cache Control Protocol defines the communication between the cache engine and the router. Using the Web Cache Control Protocol, the router directs only Web requests to the cache engine (rather than to the intended server). The router also determines cache engine availability, and redirects requests to new cache engines as they are added to an installation.

The Cisco cache engine is a single-purpose network appliance that stores and retrieves content using highly optimized caching and retrieval algorithms. (See Figure 49-3.)

Figure 49-3 This figure shows a Cisco cache engine connected to a Cisco IOS router.



Cache Engine Operation

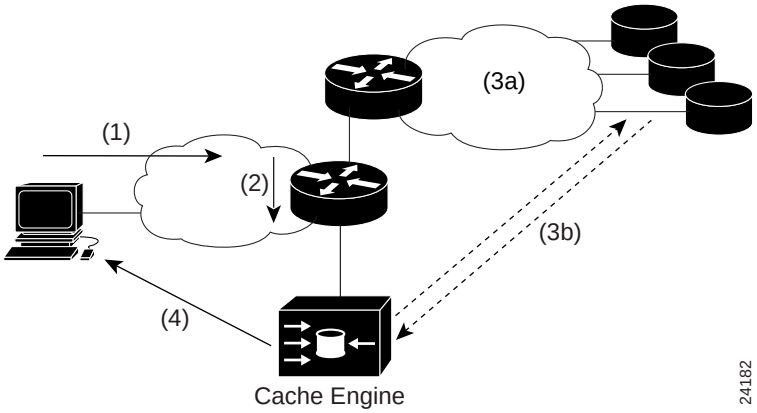
Using the Web Cache Control Protocol, the Cisco IOS router routes requests for TCP port 80 (HTTP traffic) over a local subnet to the cache engine. The cache engine is dedicated solely to content management and delivery. Because only Web requests are routed to the cache engine, no other user traffic is affected by the caching process—Web caching is done "off to the side." For non-Web traffic, the router functions entirely in its traditional role.

The cache engine works as follows (see Figure 49-4):

1. A client requests Web content in the normal fashion.
2. The router, running the Web Cache Control Protocol, intercepts TCP port 80 Web traffic and routes it to the cache engine. The client is not involved in this transaction, and no changes to the client or browser are required.
3. If the cache engine does not have the requested content, it sends the request to the Internet or intranet in the normal fashion. The content returns to, and is stored at, the cache engine.

4 The cache engine returns the content to the client. Upon subsequent requests for the same content, the cache engine fulfills the request from local storage.

Figure 49-4 This figure provides an overview of the operation of the cache engine.



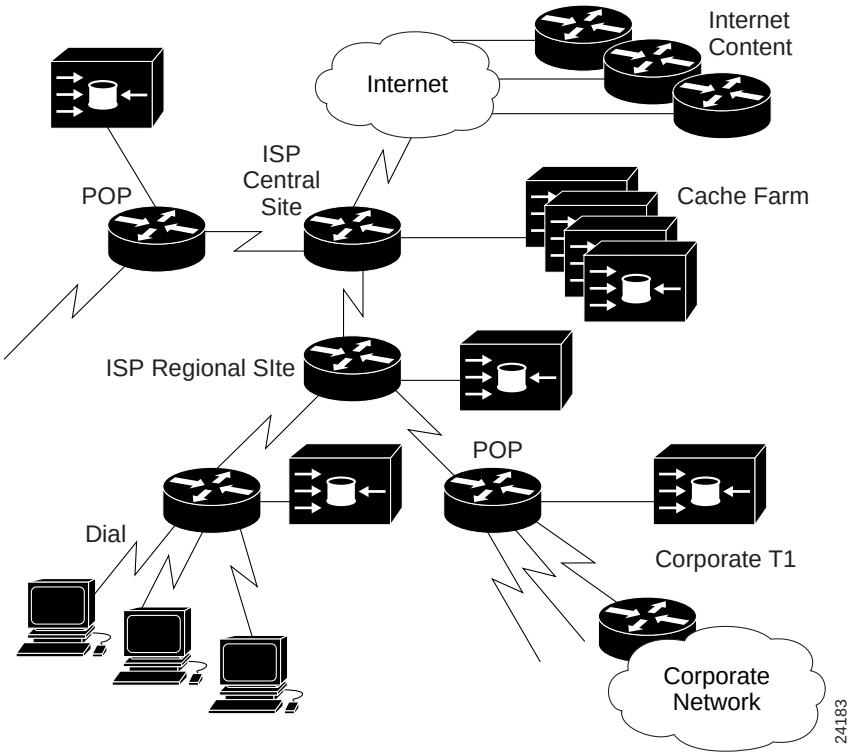
Transparency

Because the router redirects packets destined for Web servers to the cache engine, the cache engine operates transparently to clients. Clients do not need to configure their browsers to be in proxy server mode. This is a compelling feature for ISPs and large enterprises, for whom uniform client configuration is extremely expensive and difficult to implement. In addition, the operation of the cache engine is transparent to the network—the router operates entirely in its normal role for non-Web traffic. This transparent design is a requirement for a system to offer networkwide scalability, fault tolerance, and fail-safe operation.

Hierarchical Use

Because the Cisco cache engine is transparent to the user and to network operation, customers can place cache engines in several network locations in a hierarchical fashion. For example, if an ISP places a large cache farm at its main point of access to the Internet, then all its points of presence (POPs) benefit. (See Figure 49-5.) Client requests hit the cache farm and are fulfilled from its storage. To further improve service to clients, the ISP should deploy cache farms at its POPs. Then, when the client at a POP accesses the Internet, the request is diverted to the POP cache farm. If the POP cache farm is unable to fulfill the request from local storage, it makes a normal Web request. This request is routed to the cache farm at the main access point. If the request is filled by that cache farm, the traffic on the main Internet access link is avoided, the Web servers experience lower demand, and the user still experiences improved performance. As shown in Figure 49-6, enterprise networks can apply this architecture to benefit in the same ways.

Figure 49-5 This figure shows example of how to perform a hierarchical implementation of cache engines (ISP).



High Performance

The cache engine uses thread-oriented, real-time file system management and networking stack software designed solely for its role as a caching and retrieval system, eliminating the file fragmentation and long directory searches associated with general-purpose file system management design. The cache engine's secure, real-time, embedded operating system has none of the process context overhead of general-purpose operating systems such as UNIX or Windows NT; this overhead slows file access and adds to the communications load. General-purpose operating systems cannot scale to large numbers of simultaneous processes in a graceful fashion—this is especially true of a shared network cache system that can have many thousands of simultaneous, short-lived transactions. The result is an extremely high-performance, scalable cache engine.

Figure 49-6 This figure shows an example of how to perform a hierarchical implementation of cache engines (enterprise).